



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/755,381	01/05/2001	Evelyn Duesterwald	10990963-1	5203

22879 7590 06/24/2005

HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER

YIGDALL, MICHAEL J

ART UNIT PAPER NUMBER

2192

DATE MAILED: 06/24/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

09/755,381

Applicant(s)

DUESTERWALD ET AL.

Examiner

Michael J. Yigdoll

Art Unit

2192

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 05 April 2005.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 21-28, 30-39, 41 and 42 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 21-28, 30-39, 41 and 42 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. _____.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) ☐ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____
- 4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: _____

DETAILED ACTION

1. Applicant's amendment and response filed on April 5, 2005 has been fully considered.

Claims 21-28, 30-39, 41 and 42 are pending.

Response to Arguments

2. Applicant's arguments have been fully considered but they are not persuasive.

Applicant contends that although Srivastava discloses performing a liveliness analysis, Srivastava in no way discloses or suggests storing the information derived from the analysis (Applicant's remarks, page 9, second paragraph).

However, Srivastava expressly discloses that the information is stored in the LIVEIN(B) and LIVEOUT(B) sets (see, for example, column 10, lines 1-4). Each pass of the liveliness analysis updates the information stored in the LIVEIN(B) and LIVEOUT(B) sets (see, for example, column 9, lines 37-43), and thus the liveliness information must be saved or stored from one pass to the next.

Applicant further contends that there is no disclosure or suggestion in Click as to how storing the information in an epilog would provide the benefits of increased flexibility and more quickly executed code. Applicant acknowledges that Click discloses this motivation for including a register allocator that generates prologs and epilogs, but contends that neither Srivastava nor Click suggests storing liveliness information in the prologs or epilogs or provides a reason or motivation for doing so (Applicant's remarks, page 9, third paragraph).

However, Click discloses that the prologs and epilogs include code for the calling convention (see, for example, column 6, lines 38-49). The calling convention specifies how the

Art Unit: 2192

registers are used when procedures are called (see, for example, column 1, line 54 to column 2, line 7). Click discloses generating the prologs and epilogs with a register allocator so as to increase the flexibility of the code and enable the code to execute more quickly (see, for example, column 6, lines, 21-25), as acknowledged by Applicant. Indeed, one function of the register allocator is to store information about the live ranges of the registers in the calling convention code, or in other words, in the prolog and epilog code (see, for example, column 7, lines 18-32). Thus, Click discloses storing liveness information in the prologs and epilogs with a register allocator that provides increased flexibility and more quickly executed code.

One of ordinary skill in the art would have been motivated to increase the flexibility and execution speed of the code. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Srivastava with the register allocator taught by Click, such that the liveness information of Srivastava is stored in a prolog and epilog, so as to provide greater flexibility and faster execution.

Applicant contends that although Srivastava discloses generating liveness information, there is nothing in Srivastava that discloses or suggests anything about live ranges of variables (Applicant's remarks, page 10, last paragraph).

However, as Applicant acknowledges, the liveness information stored by Srivastava identifies variables that are live before (or at the start of) a block and variables that are live after (or at the end of) the block (see, for example, column 8, lines 6-12). The information represents the liveness of variables across the whole program and identifies for which blocks in the program the variables are live (see, for example, FIG. 4 and column 8, lines 13-25 and 54-61). In other words, the liveness information represents the live ranges of the variables.

Applicant further contends that although Click discloses the use of register masks to assist in determining live ranges (Applicant's remarks, page 11, second paragraph), Click does not disclose or suggest generating a first register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is assigned in an instruction identified in the first code fragment (Applicant's remarks, page 11, third paragraph), and likewise that Click fails to disclose or suggest generating a second register mask, the second register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is identified as being assigned before being read in a second code fragment (Applicant's remarks, page 11, fourth paragraph).

Specifically, Applicant contends that Click discloses setting a bit if the register is valid, rather than if the respective register is assigned and identified as being possibly live in a first code fragment, and that there is nothing in Click that discloses or suggests that the validity of a register has anything to do with whether the respective register is assigned and identified as being possibly live in a first code fragment (Applicant's remarks, page 11, third paragraph). Applicant similarly contends that there is nothing in Click that discloses or suggests that the validity of a register has anything to do with whether the respective register is assigned before being read in a second code fragment (Applicant's remarks, page 11, fourth paragraph).

However, it is Srivastava who discloses identifying an instruction that assigns a register that is possibly live in a first code fragment (see, for example, column 10, lines 18-22), and identifying a register that is assigned before being read in a second code fragment (see, for example, column 10, lines 23-25), as set forth in the previous Office action. Such analysis

Art Unit: 2192

provides liveliness information that represents the live ranges of variables, as noted above.

Likewise, the register masks of Click are used in determining the live ranges of variables, as acknowledged by Applicant. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement Srivastava with register masks such as those taught by Click.

Click discloses that the register mask is associated with a variable, and that each bit position in the register mask corresponds to a register and is set if that register is valid with respect to the variable (see, for example, column 7, lines 50-53). The intersection of register masks identifies registers that are used by more than one value (see, for example, column 8, lines 20-22). Thus, a valid register in Click is a register that may be assigned a value.

As noted above, Srivastava discloses identifying an instruction that assigns a register that is possibly live in a first code fragment. In terms of Click, therefore, this register is valid, and the bit at the respective position in the first register mask would be set. Likewise, Srivastava discloses identifying a register that is assigned before being read in a second code fragment. Again, in terms of Click, this register is valid, and the bit at the respective position in the second register mask would be set.

Therefore, Srivastava in view of Click discloses generating a first register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is assigned in an instruction identified in the first code fragment, and further discloses generating a second register mask, the second register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit

Art Unit: 2192

at a position is set if the respective register is identified as being assigned before being read in a second code fragment.

Moreover, the register masks of Click are not unrelated to the register masks of claim 27 as Applicant contends (Applicant's remarks, page 11, third paragraph). Click discloses that the register masks include a plurality of bit positions, and that each bit position corresponds to a register (see, for example, column 7, lines 50-53). One or two bits may be set depending on whether the registers represent single precision or double precision values, respectively (see, for example, column 7, lines 60-64). The number of registers represented in the register mask, however, is not limited to merely one or two. In fact, the size of the register mask is partly determined by the number of registers associated with the processor (see, for example, column 7, lines 53-55). In one example, the register mask includes 96 bit positions (see, for example, column 7, lines 41-47).

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

4. Claims 21-24, 27, 28, 30-35, 38, 39, 41 and 42 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 5,999,737 to Srivastava (art of record, "Srivastava") in view of U.S. Pat. No. 6,408,433 to Click, Jr. et al. (art of record, "Click").

With respect to claim 21 (currently amended), Srivastava discloses a method for removing dead code in code fragments of a program (see, for example, column 3, lines 6-12, which shows a method for removing dead code, and column 10, lines 9-46, which shows an instance of such dead code removal), comprising:

(a) identifying each instruction assigning a register that is possibly live for each exit in a first code fragment (see, for example, procedure or code fragment 105 in FIG. 6, and column 10, lines 18-22, which shows identifying an instruction such as block 182 defining or assigning a register R1 that is possibly live for an exit at block 185).

Although Srivastava discloses storing information corresponding to the instructions (see, for example, column 10, lines 1-4), Srivastava does not expressly disclose:

(b) storing, in an epilog associated with each exit of the first code fragment, information corresponding to each instruction identified for the corresponding exit.

However, Click discloses generating prologs and epilogs that include code for the calling convention (see, for example, column 6, lines 38-49). The calling convention specifies how the registers are used when procedures are called (see, for example, column 1, line 54 to column 2, line 7). Click discloses generating the prologs and epilogs with a register allocator so as to increase the flexibility of the code and enable the code to execute more quickly (see, for example, column 6, lines, 21-25). The register allocator stores information about the live ranges of the registers in the calling convention code, i.e. in the prolog and epilog code (see, for example, column 7, lines 18-32).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Srivastava with the register allocator taught by Click,

Art Unit: 2192

such that the information of Srivastava is stored an epilog associated with each exit of the first code fragment, so as to provide greater flexibility and faster execution.

Srivastava further discloses:

(c) identifying each register that is assigned before being read in a second code fragment having a first entry (see, for example, procedure or code fragment 104 and entry 174 in FIG. 6, and column 10, lines 23-25, which shows identifying a register R1 that is restored or assigned before being read);

(d) at a time when linking the first exit from the first code fragment to the first entry in the second code fragment, comparing the registers in the instructions identified as being possibly live in the first code fragment with the identified registers in the second code fragment (see, for example, column 10, lines 15-28, which shows comparing the registers, and column 2, lines 32-41, which shows operating at link time); and

(e) eliminating an instruction in the first code fragment based on the comparison (see, for example, column 10, lines 28-37, which shows eliminating instructions in procedure or code fragment 105 based on the comparison of registers).

With respect to claim 22 (previously presented), the rejection of claim 21 is incorporated, and Srivastava further discloses the limitation wherein an instruction identified in the first code fragment is eliminated if the register assigned in the identified instruction matches a register identified in the second code fragment (see, for example, column 10, lines 23-32, which shows that an instruction in procedure or code fragment 105 is eliminated if the registers match).

With respect to claim 23 (previously presented), the rejection of claim 21 is incorporated, and Srivastava further discloses the limitation wherein an instruction assigning a register is possibly live if there is an exit before the register is reassigned and the exit is before the register is read (see, for example, column 10, lines 18-22, which shows that an instruction assigning a register such as block 184 is possibly live if there is an exit at block 185 before register R1 is reassigned or read).

With respect to claim 24 (previously presented), the rejection of claim 21 is incorporated, and Srivastava further discloses the limitation wherein an instruction assigning a register is possibly live if the register is not read subsequently in the first code fragment (see, for example, column 10, lines 18-22, which shows that an instruction assigning a register such as block 183 is possibly live if register R2 is not read subsequently in the procedure or code fragment).

With respect to claim 27 (currently amended), Srivastava discloses a method for removing dead code in code fragments of a program (see, for example, column 3, lines 6-12, which shows a method for removing dead code, and column 10, lines 9-46, which shows an instance of such dead code removal), comprising:

(a) identifying each instruction assigning a register that is possibly live for a first exit in a first code fragment (see, for example, procedure or code fragment 105 in FIG. 6, and column 10, lines 18-22, which shows identifying an instruction such as block 182 defining or assigning a register R1 that is possibly live for an exit at block 185).

Although Srivastava discloses determining the live ranges of variables (see, for example, column 8, lines 6-25 and 54-61), Srivastava does not expressly disclose:

(b) generating a first register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is assigned in an instruction identified in the first code fragment.

However, Click discloses a register mask associated with a variable that includes a plurality of bit positions, wherein each bit corresponds to a register and is set if that register is valid with respect to the variable (see, for example, column 7, lines 50-53). The register masks are used in determining the live ranges of variables (see, for example, column 8, lines 15-20). Click discloses that the intersection of register masks identifies registers that are used by more than one value (see, for example, column 8, lines 20-22). Thus, a register is considered valid and its respective bit in the register mask is set if the register may be assigned a value.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Srivastava with register masks, such as taught by Click, so as to provide an additional means by which to determine live ranges.

As noted above, Srivastava discloses identifying each instruction assigning a register that is possibly live for a first exit in a first code fragment. Therefore, Srivastava in view of Click discloses generating a first register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is assigned in an instruction identified in the first code fragment.

Srivastava in view of Click further discloses:

(c) identifying each register that is assigned before being read in a second code fragment having a first entry (see, for example, Srivastava, procedure or code fragment 104 and entry 174

Art Unit: 2192

in FIG. 6, and column 10, lines 23-25, which shows identifying a register R1 that is restored or assigned before being read);

(d) generating a second register mask, the second register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is one of the identified registers in the second code fragment (see the explanation for part (b) above);

(e) at a time when linking the first exit from the first code fragment to the first entry in the second code fragment, comparing the registers in the instructions identified as being possibly live in the first code fragment with the identified registers in the second code fragment (see, for example, Srivastava, column 10, lines 15-28, which shows comparing the registers, and column 2, lines 32-41, which shows operating at link time); and

(f) eliminating an instruction in the first code fragment based on the comparison (see, for example, Srivastava, column 10, lines 28-37, which shows eliminating instructions in procedure or code fragment 105 based on the comparison of registers).

With respect to claim 28 (previously presented), the rejection of claim 27 is incorporated, and Srivastava in view of Click further discloses the limitation wherein said eliminating step includes eliminating an instruction for assigning a register in the first code fragment if the positions corresponding to the register in the first and second register masks are both set (see, for example, Srivastava, column 10, lines 28-37, which shows eliminating the instructions after determining the live ranges, and Click, column 8, lines 15-22, which shows determining the live ranges based on the intersection of the register masks).

With respect to claim 30 (currently amended), the rejection of claim 21 is incorporated, and Srivastava further discloses:

(a) identifying each register that is assigned before being read after each entry in the second code fragment (see, for example, procedure or code fragment 104 and entry 174 in FIG. 6, and column 10, lines 23-25, which shows identifying a register R1 that is restored or assigned before being read).

Although Srivastava discloses storing information corresponding to the instructions (see, for example, column 10, lines 1-4), Srivastava does not expressly disclose:

(b) storing, in a prolog associated with each entry of the second code fragment, information corresponding to each register identified for the corresponding entry.

However, Click discloses generating prologs and epilogs that include code for the calling convention (see, for example, column 6, lines 38-49). The calling convention specifies how the registers are used when procedures are called (see, for example, column 1, line 54 to column 2, line 7). Click discloses generating the prologs and epilogs with a register allocator so as to increase the flexibility of the code and enable the code to execute more quickly (see, for example, column 6, lines, 21-25). The register allocator stores information about the live ranges of the registers in the calling convention code, i.e. in the prolog and epilog code (see, for example, column 7, lines 18-32).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Srivastava with the register allocator taught by Click, such that the information of Srivastava is stored in a prolog associated with each entry of the second code fragment, so as to provide greater flexibility and faster execution.

With respect to claim 31 (previously presented), the rejection of claim 30 is incorporated, and Srivastava further discloses the limitation wherein the information in the corresponding epilogs of the exits in the first code fragment and the information in the corresponding prologs of the entries in the second code fragment are stored prior to the linking of the first exit from the first code fragment to the first entry in the second code fragment (see, for example, column 10, lines 38-46, which shows that the information is stored prior to removing dead code at link time).

With respect to claim 32 (currently amended), the claim recites a computer readable medium that corresponds to the method of claim 21. Srivastava discloses a computer readable medium operable on a computer for removing dead code in code fragments of a program (see, for example, column 3, lines 6-12 and 57-63). The limitations recited in the claim are analogous to those of claim 21 (see the rejection of claim 21 above).

With respect to claims 33-35 (previously presented), the limitations recited in the claims are analogous to those of claims 22-24, respectively (see the rejections of claims 22-24 above).

With respect to claim 38 (currently amended), the claim recites a computer readable medium that corresponds to the method of claim 27. Srivastava discloses a computer readable medium operable on a computer for removing dead code in code fragments of a program (see, for example, column 3, lines 6-12 and 57-63). The limitations recited in the claim are analogous to those of claim 27 (see the rejection of claim 27 above).

With respect to claims 39 (previously presented), the limitations recited in the claim are analogous to those of claim 28 (see the rejection of claim 28 above).

With respect to claims 41 (currently amended) and 42 (previously presented), the limitations recited in the claims are analogous to those of claims 30 and 31, respectively (see the rejections of claims 30 and 31 above).

5. Claims 25 and 36 are rejected under 35 U.S.C. 103(a) as being unpatentable over Srivastava in view of Click, as applied to claims 21 and 32 above, respectively, and further in view of U.S. Pat. No. 6,112,025 to Mulchandani et al. (art of record, "Mulchandani").

With respect to claim 25 (previously presented), the rejection of claim 21 is incorporated, but Srivastava in view of Click does not expressly disclose the limitation wherein eliminating the instruction comprises overwriting the instruction with a NOP.

However, Mulchandani discloses replacing an instruction with a NOP (see, for example, column 5, lines 30-37) so as to prevent the unwanted effects of executing the instruction (see, for example, column 5, lines 42-50).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to eliminate the instruction by overwriting the instruction with a NOP, such as taught by Mulchandani, so as to prevent the unwanted effects of executing the instruction.

With respect to claim 36 (previously presented), the limitations recited in the claim are analogous to those of claim 25 (see the rejection of claim 25 above).

6. Claims 26 and 37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Srivastava in view of Click, as applied to claims 21 and 32 above, respectively, and further in view of U.S. Pat. No. 6,041,179 to Bacon et al. (art of record, "Bacon").

With respect to claim 26 (previously presented), the rejection of claim 21 is incorporated, but Srivastava in view of Click does not expressly disclose the limitation wherein eliminating the instruction comprises compacting the surrounding instructions to delete the eliminated instruction.

However, Bacon discloses compacting code in a program by linking only the live procedures and not including those determined to be dead (see, for example, column 11, lines 9-13), so as to reduce the code size.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to eliminate the instruction by compacting the surrounding instructions, such as taught by Bacon, so as to reduce the code size.

With respect to claim 37 (previously presented), the limitations recited in the claim are analogous to those of claim 26 (see the rejection of claim 26 above).

Conclusion

7. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37

Art Unit: 2192

CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (571) 272-3707.

The examiner can normally be reached on Monday through Friday from 7:30am to 4:00pm.

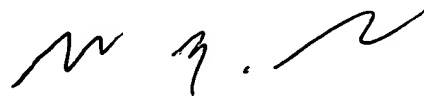
If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall
Examiner
Art Unit 2192

mjy



WEI Y. ZHEN
PRIMARY EXAMINER